Tarea 3
Bases de Datos
"Triunfar en los negocios"

Marco Espinoza
marco.espinozaco@usm.cl

Juan Mira
juan.mirag@usm.cl

Javier Peralta
javier.peraltae@usm.cl

Rodrigo Munita
rmunita@usm.cl

Ricardo Lorca
ricardo.lorca@sansano.usm.cl

Cecilia Reyes
cecilia.reyes@usm.cl

Sebastián Gallardo
sebastian.gallardod@usm.cl

14 de Junio, 2022

1. Introducción

Tras la implementación del sistema de música, el Señor Burns pudo notar su eficiencia y el aumento considerable de usuarios. Por ello, desde el área de finanzas se ha decidido ampliar el proyecto, añadiéndole más funciones a la plataforma del lado gerencial, lo cual implica también cambios en el modelo de la base de datos. Como dichos cambios serán meramente para la toma de decisiones, se deberá implementar una nueva base de datos con dichos cambios. Se adjunta el modelo siguiente:

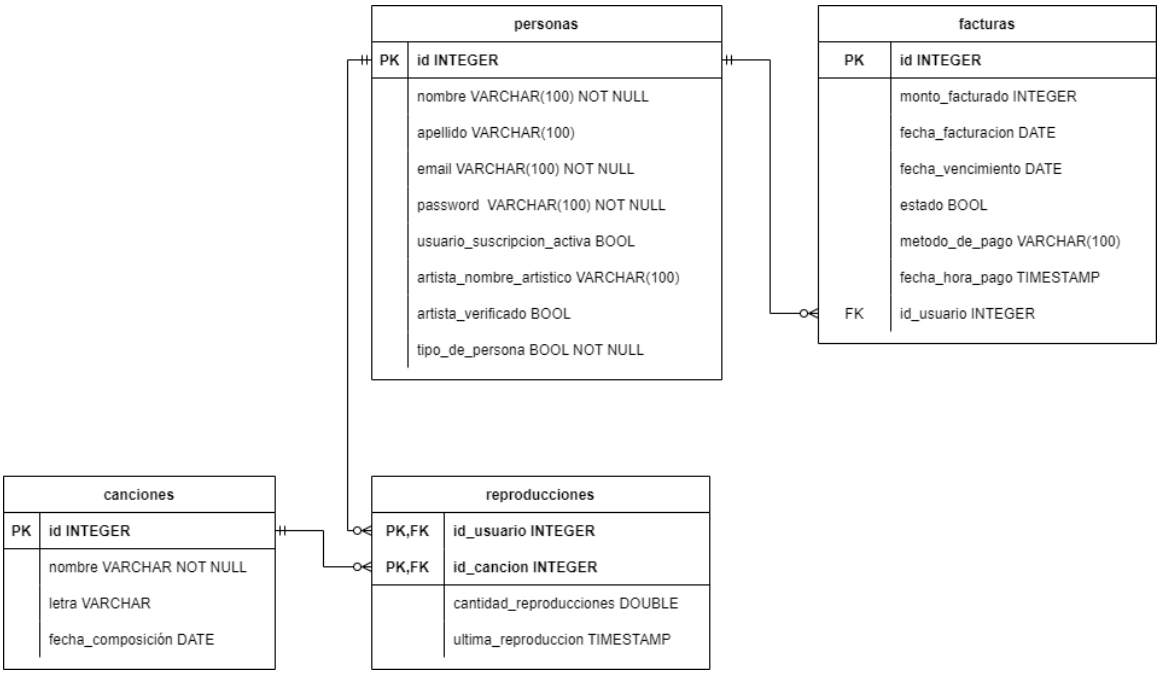


Figura 1: Modelo propuesto

En el modelo se aprecian las tablas "personas" y "canciones", dichas tablas son del modelo del sistema musical que ustedes implementaron en la Tarea 2 (con la herencia ya resuelta) y se añadieron meramente para darle sentido a las relaciones. Por lo cual, en esta tarea ustedes van a trabajar mayormente con las tablas de "reproducciones" y "facturas".

En la tabla "facturas", el atributo "estado" indica si la factura se encuentra pagada o no:

- True: Factura pagada,
- False: Factura no pagada.

2. Instrucciones Técnicas

2.1. Implementación de la API REST

Se solicita implementar los siguientes puntos en una API REST, utilizando el stack de tecnologías señalado en ayudantías (ver condiciones y stack de tecnologías):

Hint: Recordar que algunos endpoints se pueden resolver directamente con ORM, y otros con consultas sql personalizadas (a través de SQLAlchemy).

1. Implementar un CRUD para cada tabla del modelo.
   - Cada tabla debe tener cuatro endpoints, o sea un endpoint por cada operación CRUD: CREATE, READ, UPDATE, DELETE.
   - Para la tabla personas, no es necesario que se le aplique una función hash a la contraseña.

2. Implementar un endpoint que indique si el usuario es moroso (facturas impagas que ya vencieron).

   - Se considera una factura vencida si ha transcurrido un día desde la fecha de vencimiento. Por ejemplo, si una factura vence el 06/06/2022 se considera vencida a contar del día 07/06/2022 (inclusive).
     Hint: Utilice la fecha actual de ejecución revisar si la factura está vencida
   - Se debe utilizar el ID del usuario para identificarlo, se recomienda pasar dicho ID a través de una query parameter.
   - En caso de que el usuario no tenga deuda, indicar con un mensaje que el usuario no tiene facturas pendientes.
   - Si el usuario tiene deuda, indicar con un mensaje que tiene facturas vencidas junto con una lista de las facturas vencidas.


```
{
  "mensaje": "El usuario tiene facturas vencidas.",
  "facturas": [
    {
      "id_factura": 1,
      "monto_facturado": 10000,
      "fecha_facturacion": "01/01/2022",
      "fecha_vencimiento": "15/01/2022"
    },
    {
      "id_factura": 2,
      "monto_facturado": 10000,
      "fecha_facturacion": "01/02/2022",
      "fecha_vencimiento": "15/02/2022"
    }
  ]
}
```


Figura 2: Output de ejemplo

3. Implementar un endpoint que indique la cantidad de personas morosas (usuarios con facturas vencidas impagas, utilizar los mismos criterios mencionados anteriormente) y la cantidad de dinero total que le deben a la compañía.


```
{
  "qty_personas": 300,
  "qty_dinero": 50000
}
```


Figura 3: Output de ejemplo

4. Implementar un endpoint que indique los ingresos de la compañía de los últimos 31 días. O sea, la suma de todas las facturas pagadas dentro de los últimos 31 días.

   - *Hint: se debe utilizar el atributo "fecha_hora_pago" de la tabla "facturas".*

- *Hint 2: Utilice la fecha actual de ejecución revisar que se encuentre dentro de los ultimos 31 días*


```
{
  "qty_dinero": 5000000
}
```


Figura 4: Output de ejemplo

5. Implementar un endpoint que retorne el TOP 10 de canciones más escuchadas por un usuario, cuyas reproducciones deben estar en orden descendente.
   - Si hay menos de 10 canciones, se debe retornar las que haya.
   - Si el usuario no ha escuchado ninguna canción, retornar un mensaje indicándolo.
   - Se debe utilizar el ID del usuario para identificarlo, se recomienda pasar dicho ID a través de una query parameter.


```
{
  "top_ten": [
    {
      "id_cancion": 1,
      "nombre_cancion": "Never Gonna Give You Up",
      "reproducciones": 102
    },
    {
      "id_cancion": 3,
      "nombre_cancion": "The Hero",
      "reproducciones": 54
    }
  ]
}
```


Figura 5: Output de ejemplo

6. Implementar un endpoint que retorne el TOP 10 de canciones más escuchadas de forma global. Es similar al endpoint anterior, ya que debe retornar el mismo tipo de output, pero de todos los usuarios.

2.2. Documentación de la API REST

- Se solicita documentar cada endpoint de la API REST en Postman y publicar dicha documentación, generando una URL.
- Dicha URL se debe añadir al informe de la tarea (ver detalles del informe).
- Se solicita seguir la estructura vista en ayudantía (ver PPT de "Documentación en Postman").

2.3. Implementación del Reporte

Se solicita crear un reporte de inteligencia de negocios utilizando Microsoft Power BI Desktop, que considere los siguientes puntos:

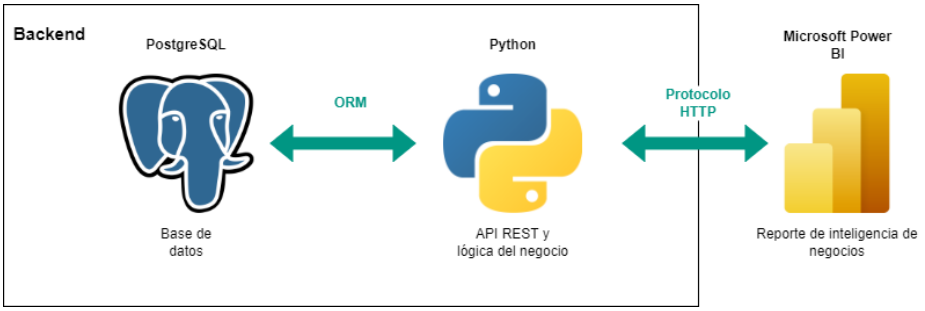
1. Un visualizador del tipo tarjeta que indique la cantidad de personas morosas, o sea la cantidad de personas que tienen al menos una factura vencida e impaga.
2. Un visualizador del tipo tarjeta que indique la cantidad de dinero total que le deben a la compañía (suma de todos los montos de las facturas vencidas e impagas de todos los usuarios).
3. Un visualizador del tipo tarjeta que indique los ingresos de los últimos 31 días de la compañía.
4. Un visualizador del tipo tabla, que muestre una lista con los nombres de las 10 canciones más reproducidas a nivel global junto con la cantidad de reproducciones. Las canciones deben estar ordenadas por reproducciones de forma descendente.

Se deben utilizar los endpoints de la API REST para alimentar el reporte y los visualizadores.

El diseño es una parte fundamental dentro de esta instancia, puesto que es un reporte y debe ser presentado y comprendido por una persona externa, por lo tanto, también se evaluara la creatividad, orden y diseño.

2.4. Arquitectura

Para que tengan un mejor entendimiento de lo solicitado, se adjunta el siguiente diagrama que muestra la interacción de las diversas tecnologías:



3. Condiciones de Entrega

3.1. Informe

Dentro del mundo del desarrollo, nada está 100 % estandarizado, y las ideas e implementaciones varían de desarrollador en desarrollador. Es por esto que, independiente de cómo o qué implementen en su entrega, les solicitamos que adjunten a esta un **Informe**, el cual detallará todas sus consideraciones, suposiciones, detalles de uso y notas respecto al sistema desarrollado. Como mínimo se solicita que el informe:

- Debe incluir los **datos personales de cada alumno** de su grupo, junto con su número de ROL USM.
- *(Opcional) Si lo desean, pueden adjudicarse un "Nombre de Grupo" y utilizarlo como marca o empresa desarrolladora de la API.*
- Debe incluir **todos los supuestos y consideraciones** que uds. tomaron en cuenta al desarrollar la entrega.
- Debe incluir un enlace al repositorio GIT de la tarea (idealmente el mismo repositorio de las entregas anteriores). El repositorio debe ser privado, es por ello que deben dar permisos a los ayudantes.
- Debe incluir un enlace a la documentación pública de su API REST en Postman.
- En una sección final, solicitamos que informen acerca de sus **dificultades** con esta implementación; en concreto, que puedan reportar cuáles tareas fueron las más difíciles (o largas) en su desarrollo. Puede variar desde instalación de tecnologías, implementación del modelo, proceso creativo, desarrollo, etc.
- En esta misma última sección, solicitamos que reporten un estimado de cuánto **tiempo** les tomó el desarrollo de la tarea completa (valores por persona), buscando incluir todo momento que gastaron en ella.
    - *Estimadores Recomendados: Análisis del Enunciado, Modificaciones e Implementación del modelo, Diseño del sistema, Documentaciones, Pruebas Finales, etc.*

El informe debe venir en formato **PDF**, e incluir las secciones detalladas anteriormente.

Nombre	**Github**
Marco	mavi3
Juan	juan-mira
Javier	Lord-Ureiva
Ricardo	rilodeh
Rodrigo	rgomunita

Cuadro 1: Usuarios de Github

3.2. Implementación

Respecto a la implementación de los requisitos, se solicita:

- El código debe estar **ordenado**, es decir, identado, estructurado y lo más comentado posible.
- Se debe utilizar ORM para manejar la capa de persistencia.
- Se debe utilizar correctamente el protocolo HTTP, esto incluye el buen uso de los métodos y códigos de estado.
- Se debe mantener la estructura de los archivos lo más ordenado posible.

4. Condiciones Generales

- La fecha de entrega es el **Miércoles 20 de Julio del 2022 a las 23:59PM**.
  - Se aplicará un descuento lineal para tareas atrasadas hasta las **23:59 PM** del día siguiente.
- Las tecnologías a utilizar para el desarrollo de la plataforma son:
  - **Python, Flask y SQL Alchemy:** Para implementar la API REST, la lógica del lado del servidor, las conexiones a la base de datos y la capa de persistencia.
  - **Postman:** Para documentar la API REST.
  - **PostgreSQL 14:** La base de datos asociada al sistema.
  - **Microsoft Power BI Desktop:** Para la creación y diseño de reportes de inteligencia de negocios.
- Su entrega debe incluir el **informe** (nombre de archivo **InformeGrupoX.pdf**), un **dump** de su base de datos (nombre de dump **DumpGrupoX.sql**) y el **reporte** realizado en Microsoft Power BI Desktop (nombre del reporte **ReporteGrupoX.pbix**). Todo esto en un comprimible de nombre **Tarea3-GrupoX.zip**, donde X es el número de su grupo.
- Es muy importante incluir el enlace al repositorio GIT en el informe, puesto que se evaluaran los commits realizados por cada integrante del grupo.
- Integrante que no es incluido en el informe y/o no participa en el desarrollo de la tarea tendrá **Nota 0**.
- La entrega de los archivos es por el medio Aula, sólo uno de los integrantes del grupo debe realizar la entrega.

5. Recomendaciones

- Aún en un gran número de tecnologías usadas en el oficio de desarrollo de software, es común que la instalación de tecnologías sea un paso difícil de realizar. Sin embargo, les recomendamos que, ante cualquier fallo o impedimento, **busquen en internet si alguien ha tenido el mismo problema**. Muchas veces horas de peleas se ahorran con un milagroso post en Stackoverflow del 2014.
- Finalmente, consejo general de desarrollador: Si quieres saber cómo hacer X con Y tecnología, googlea "How to do X with Y" (o en español si les complica el Inglés) y revisa los resultados.